Unanswered Questions User Study		
1. How to read an excel dataframe from a private GitHub repository using pandas? (57164629)	SE	If you want only the remote URL, or if your are not connected to a network that can reach the remote repo: git config --get remote.origin.url If you require full output and you are on a network that can reach the remote repo where the origin resides : git remote show origin When using git clone (from GitHub, or any source repository for that matter) the default name for the source of the clone is "origin". Using git remote show will display the information about this remote name. The first few lines should show: … If you want to use the value in the script, you would use the first command listed in this answer.
	Google	You should provide URL to raw content. Try using this: import pandas as pd url = 'https://raw.githubusercontent.com/lukes/ISO-3166-Countries-with-Regional-Codes/master/all/all.csv' df = pd.read_csv(url, index_col=0) print(df.head(5))
	Ours	This can be done with Pandas sudo pip install pandas By convention, it's abbreviated as pd when imported: import pandas as pd Once imported, use Panda's read_excel attribute to load the excel file as a Pandas Dataframe: df = pd.read_excel('path/to/xlxs_file/') df.head() Read more on Pandas here.
2. How do i read docx and pdf file from hdfs using python command line (57165142)	SE	Look at the subprocess module in the standard library: import subprocess subprocess.run(["ls", "-l"]) The advantage of subprocess vs. system is that it is more flexible (you can get the stdout, stderr, the "real" status code, better error handling, etc...). The official documentation recommends the subprocess module over the alternative os.system(): The subprocess module provides more powerful facilities for spawning new processes and retrieving their results; using that module is preferable to using this function [os.system()]. The Replacing Older Functions with the subprocess Module section in the subprocess documentation may have some helpful recipes.For versions of Python before 3.5, use call: import subprocess subprocess.call(["ls", "-l"])

	Google	The docx is a zip file containing an XML of the document. You can open the zip, read the document and parse data using ElementTree. The advantage of this technique is that you don't need any extra python libraries installed.
		import zipfile
		import xml.etree.ElementTree
		WORD_NAMESPACE = '{http://schemas.openxmlformats.org/wordprocessingml/2006/main}'
		PARA = WORD_NAMESPACE + 'p'
		TEXT = WORD_NAMESPACE + 't'
		TABLE = WORD_NAMESPACE + 'tbl'
		ROW = WORD_NAMESPACE + 'tr'
		CELL = WORD_NAMESPACE + 'tc'
		with zipfile.ZipFile('<path to docx file>') as docx:
		  tree = xml.etree.ElementTree.XML(docx.read('word/document.xml'))
		for table in tree.iter(TABLE):
		  for row in table.iter(ROW):
		  for cell in row.iter(CELL):
		  print ''.join(node.text for node in cell.iter(TEXT))

	Ours	you can use this code to read the file and fill data
		with open("file.txt", "r") as f:
		  lines = f.readlines()
		  data = [line.split("\n")[0].split(";") for line in lines]

3 How to copy folder from server to client in python (57165374)	SE	Look at the subprocess module in the standard library: import subprocess subprocess.run(["ls", "-l"]) The advantage of subprocess vs. system is that it is more flexible (you can get the stdout, stderr, the "real" status code, better error handling, etc…). The official documentation recommends the subprocess module over the alternative os.system(): The subprocess module provides more powerful facilities for spawning new processes and retrieving their results; using that module is preferable to using this function [os.system()]. The Replacing Older Functions with the subprocess Module section in the subprocess documentation may have some helpful recipes.For versions of Python before 3.5, use call: import subprocess subprocess.call(["ls", "-l"])
	Google	You need to use SSH, SCP, or SFTP to transfer files from host to host I do this a lot and like to use SSH and SCP. You can run and SSH server on your windows machine using OpenSSH. Here is a good set of instructions from WinSCP: https://winscp.net/eng/docs/guide_windows_openssh_server. I recommend using Paramiko for SSH with Python. Here is a good answer showing how this works with python: https://stackoverflow.com/a/38556344/634627. If you set up OpenSSH, you could also do this with SFTP, sometimes I find this is more suitable that SCP. Here is a good answer showing how that works:https://stackoverflow.com/a/33752662/634627 The trick is getting OpenSSH running on your Windows host and setting up SSH keys so your server can authenticate to your localhost.
	Ours	I don't know what you are exactly trying (and asking) but you need to use relative or absolute paths to your image instead of urls. currentpath = '/img/default/basicprof.jpg' newpathandname = '/img/{}.jpg'.format(username) os.rename(currentpath, newpathandname) #does not keep original file shutil.copyfile(currentpath, newpathandname) #keeps original file

Unanswered Questions User Study		

4. OperationalError at /result unable to open database file(57168023)	SE	If the reason you're checking is so you can do something like if file_exists: open_it(), it's safer to use a try around the attempt to open it. Checking and then opening risks the file being deleted or moved or something between when you check and when you try to open it.
		If you're not planning to open the file immediately, you can use os.path.isfile Return True if path is an existing regular file. This follows symbolic links, so both islink() and isfile() can be true for the same path.
		import os.path
		os.path.isfile(fname)
		if you need to be sure it's a file. Starting with Python 3.4, the pathlib module offers an object-oriented approach (backported to pathlib2 in Python 2.7):
		from pathlib import Path
		my_file = Path("/path/to/file")
		if my_file.is_file():
		  # file exists
		To check a directory, do:
		if my_file.is_dir():
		  # directory exists
		To check whether a Path object exists independently of whether is it a file or directory, use exists():
		if my_file.exists():
		  # path exists
		You can also use resolve(strict=True) in a try block:
		try:
		  my_abs_path = my_file.resolve(strict=True)
		except FileNotFoundError:
		  # doesn't exist
		else:
		  # exists

	Google	Primary diagnosis: SQLite is unable to open that file for some reason. Checking the obvious reasons why, and in approximate order that I recommend checking: Is the program running on the same machine as you're testing it? Is it running as you (or at least the same user as you're testing it as)? Is the disk containing /tmp full? (You're on Unix, so use df /tmp to find out.) Does the /tmp/cer directory have “odd” permissions? (SQLite needs to be able to create additional files in it in order to handle things like the commit log.) Is the unit test code still using that database? (Concurrent opens are possible with a modern-enough SQLite and when in the right filesystem — though /tmp is virtually always on the right sort of FS so it's probably not that — but it's still not recommended.) Is the development code really trying to write to that database, or is something “clever” catching you out and causing it to try to open something else? (I've been caught out by this in my code in the past; don't think it can't happen to you…) Are you using the same version of the SQLite library in the unit tests and the production code? If you're not on the same machine, it's quite possible that the production system doesn't have a /tmp/cer directory. Obvious to fix that first. Similarly, if you're on the same machine but running as different users, you're likely to have permissions/ownership problems. Disk space is another serious gotcha, but less likely. I don't think it's the last three, but they're worth checking if the more obvious deployment problems are sorted. If it's none of the above, you've hit an exotic problem and will have to report much more info (it might even be a bug in SQLite, but knowing the developers of it, I believe that to be quite unlikely).
	Ours	Check rights of user, from whom you are executing python manage.py syncdb. This user must have rights to read and write for file /opt/graphite/webapp/test.db And also to read, write and execute rights for folder /opt/graphite/webapp/ Try this commands (execute from the user, that starts python manage.py syncdb, probably it is current user) sudo chown `whoami`:`whoami` /opt/graphite/webapp/test.db # change owner sudo chmod o+rw /opt/graphite/webapp/test.db # add rights sudo chown `whoami`:`whoami` /opt/graphite/webapp/ sudo chmod o+rwx /opt/graphite/webapp/

5. How to pass variables from parent class to child class? (57169624)	SE	Variables declared inside the class definition, but not inside a method are class or static variables:
		As @millerdev points out, this creates a class-level i variable, but this is distinct from any instance-level i variable, so you could have …
		This is different from C++ and Java, but not so different from C#, where a static member can't be accessed using a reference to an instance.
		class C:
		  @staticmethod
		  def f(arg1, arg2, ...): ...
		@beidy recommends classmethods over staticmethod, as the method then receives the class type as the first argument, but I'm still a little fuzzy on the advantages of this approach over staticmethod. If you are too, then it probably doesn't matter

	Google	The variable firstname1 is a local variable. You can't access it outside its scope - the method. What you can do is pass a copy of the reference to your subclass. Since you're calling a static method, the easiest way is to pass the reference as an argument to the method call: … That said, since you're using inheritance, you might find it useful to use an instance method. Remove "static" from the method. In main(), construct an instance of the class, provide it the name, and call the method on the instance.

	Ours	Sure. This is not an uncommon pattern:
		class A(object):
		  def __init__(self, foo, bar=3):
		  self.foo = foo
		  self.bar = bar
		class B(A):
		  def __init__(self, quux=6, **kwargs):
		  super(B, self).__init__(**kwargs)
		  self.quux = queue
		B(foo=1, quux=4)
		This also insulates you a little from super shenanigans: now A's argspec can change without requiring any edits to B, and diamond inheritance is a little less likely to break.

6. Celery Worker exited prematurely: signal 9 (SIGKILL) on serializer.is_valid(57183084)	SE	Hate it when I end up answering my own question. Anywhoo, it was blocking on a join call on the Mediator component inside the WorkController (calls stop() on the Mediator component, inside stop, it joins). I got rid of the Mediator component by disabling all rate limits (should be this by default, but it's not for some reason). You can disable all rate limits with the setting: CELERY_DISABLE_ALL_RATE_LIMITS: True Hope this helps somebody else down the road too. Peace
	Google	None

Unanswered Questions User Study		
	Ours	If your celery workers are issuing print statements or using the standard python logging classes, then you should see something that looks like this in your heroku logs: 2015-04-25T00:25:00.072929+00:00 app[celeryd.1]: [2015-04-25 00:25:00,072: INFO/Beat] Scheduler: Sending due task contest-winner (app.contest_win) celeryd will be replaced by whatever you named the celery worker in your ProcFile. Mine looks like this and you can see the process label in the logging statement above: celeryd: celery -A app.celery worker -E -B --loglevel=INFO Of course, you have to set the log level too :).
7. list index out of range reading files (57183862)	SE	if not a: print("List is empty") Using the implicit booleanness of the empty list is quite pythonic.

	Google	You need to actually read the file. Some thing like this:
		def compare():
		  with open("Rules.txt") as file1:
		  for rules in file1.readlines(): // <-note readlines()
		  rule = rules.split(",")
		  break
		  with open("Members.txt") as membersFile:
		  for members in membersFile.readlines(): // <-here too
		  member = members.split(',')
		  break
		  print (rule[0] + "," +member[1] + ","+ rule[1] + "," +member[2] + ","+ rule[2] + "," + member[3])
		  if (rule[0]<=member[1]) and (rule[1]<=member[2]) and (rule[2]==member[3]):
		  print (member[0] + " is Continued")
		  else:
		  print (member[0] + " is Discontinued")
		But actually, the break statements also seem to certainly be wrong, and possibly the split(",") also - as in the comments - really need a properly formmatted sample of the input, and a better description of the expected behavior.

	Ours	A simple condition in the for loop may solve the issue.
		def change():
		  # read file into dictionary
		  with open('exchangeRate.csv', 'r') as in_file:
		  echRdr = csv.reader(in_file)
		  for row in echRdr:
		  if len(row) <= 1:
		  pass
		  else:
		  rates[row[0]] = row[1]

8. How do I send username and password(api key) to a socket using pycurl? (57187308)	SE	In Python 3 use input(): input("Press Enter to continue...") In Python 2 use raw_input(): raw_input("Press Enter to continue...") This only waits for the user to press enter though. One might want to use msvcrt ((Windows/DOS only) The msvcrt module gives you access to a number of functions in the Microsoft Visual C/C++ Runtime Library (MSVCRT)): import msvcrt as m def wait(): m.getch() This should wait for a key press. Additional info: in Python 3 raw_input() does not exist In Python 2 input(prompt) is equivalent to eval(raw_input(prompt))

	Google	In Python 3 the following will work. I am using the lower level http.client from the standard library. Also check out section 2 of rfc2617 for details of basic authorization. This code won't check the certificate is valid, but will set up a https connection. See the http.client docs on how to do that.
		from http.client import HTTPSConnection
		from base64 import b64encode
		#This sets up the https connection
		c = HTTPSConnection("www.google.com")
		#we need to base 64 encode it
		#and then decode it to acsii as python 3 stores it as a byte string
		userAndPass = b64encode(b"username:password").decode("ascii")
		headers = { 'Authorization' : 'Basic %s' % userAndPass }
		#then connect
		c.request('GET', '/', headers=headers)
		#get the response back
		res = c.getresponse()
		# at this point you could check the status etc
		# this gets the page text
		data = res.read()

	Ours	What you need is an activation URL. The activation URL contains a unique ID, a UUID perhaps, which is verified when the user clicks the activation URL. To avoid storing the user data in the database, you could store the user data in the activation URL sent to the user: import urllib data = urllib.urlencode({'name':'joe', 'password':'1234'}) activation_url = 'http://example.com/activate?%s' % data But because user information is sent in plain text this method is very insecure. The correct way to approach this is to store the user's information along with an activation flag in the database. When the user clicks the activation URL, the unique ID is verified and the activation flag is set to true enabling the new user account. Sending email in Python is pretty straight forward when you have access to a SMTP server or Gmail account. The Python docs contain some examples of sending email from Python.
9. Sklearn Pipeline/ColumnTransformer throwing ValueError: too many values to unpack (expected 2) (57190958)	SE	I had a quick look and the SVC class (and the cross validation tool) both seem to expect kernel callables to compute the whole kernel matrix at once from the full-data matrix (which makes this feature very limited I agree). Please have a look at the tests for more details:

	Google	Your brackets are in the wrong place / you are missing brackets when creating the Pipeline, should be a list of tuples:
		pipeline = Pipeline([
		  ('text_length', AverageWordLengthExtractor()),
		  ('scale', StandardScaler())
		])

	Ours	Your brackets are in the wrong place / you are missing brackets when creating the Pipeline, should be a list of tuples:
		pipeline = Pipeline([
		  ('text_length', AverageWordLengthExtractor()),
		  ('scale', StandardScaler())
		])

Unanswered Questions User Study		
10. difference between objects.create() and object.save() in django orm (23926385)	SE	Let's say you have a collection of Car objects (database rows), and each Car has a collection of Wheel objects (also rows). In other words, Car → Wheel is a 1-to-many relationship. Now, let's say you need to iterate through all the cars, and for each one, print out a list of the wheels. The naive O/R implementation would do the following: SELECT * FROM Cars; And then for each Car: SELECT * FROM Wheel WHERE CarId = ? In other words, you have one select for the Cars, and then N additional selects, where N is the total number of cars. Alternatively, one could get all wheels and perform the lookups in memory: SELECT * FROM Wheel This reduces the number of round-trips to the database from N+1 to 2. Most ORM tools give you several ways to prevent N+1 selects.
	Google	https://docs.djangoproject.com/en/stable/topics/db/queries/#creating-objects To create and save an object in a single step, use the create() method.
	Ours	https://docs.djangoproject.com/en/stable/topics/db/queries/#creating-objects To create and save an object in a single step, use the create() method.

11. How to change behaviour of isinstance in python (57191681)	SE	For dictionaries x and y, z becomes a shallowly merged dictionary with values from y replacing those from x.
		In Python 3.5 or greater: z = {**x, **y}
		In Python 2, (or 3.4 or lower) write a function:
		def merge_two_dicts(x, y):
		  z = x.copy() # start with x's keys and values
		  z.update(y) # modifies z with y's keys and values & returns None
		  return z
		…

	Google	Here's something I think does what you want. The wrap() function dynamically creates a class which is derived from the class of the obj argument passed to it, and then returns an instance of that class created from it. This assumes the class of obj supports copy construction (initialization from an instance of the same — or derived — class).
		def wrap(obj):
		  class MetaClass(type):
		  def __new__(mcls, classname, bases, classdict):
		  wrapped_classname = '_%s_%s' % ('Wrapped', type(obj).__name__)
		  return type.__new__(mcls, wrapped_classname, (type(obj),)+bases, classdict)
		  class Wrapped(metaclass=MetaClass):
		  pass
		  return Wrapped(obj)

	Ours	Not exactly clear what you need you can do something like this (for example):
		class Bar(object):
		  def __init__(self, y=None):
		  self.y = y
		class Foo(object):
		  def __init__(self, x, y):
		  self.x = x
		  self.x.y = y
		if __name__ == '__main__':
		  x = Bar()
		  y = 1
		  foo = Foo(x,y)

12. Replace dataframe values with data from another dataframe that match (57196606)	SE	Call map and pass the dict, this will perform a lookup and return the associated value for that key: In [248]: d = {112: 'en', 113: 'es', 114: 'es', 111: 'en'} df['D'] = df['U'].map(d) df Out[248]: U L D 0 111 en en 1 112 en en 2 112 es en 3 113 es es 4 113 ja es 5 113 zh es 6 114 es es
	Google	Use merge with left join and replace missing values by original values by fillna: df = df1.merge(df2, on=['ID', 'Month'], how='left', suffixes=('_','')) df['Value'] = df['Value'].fillna(df['Value_']).astype(int) df = df.drop('Value_', axis=1)
	Ours	when data has looping replacement(i.e overlaping keys and values), replacement on whole dataframe will fail. In this case, doing it column by column and concat them together. Finally, use join to adding any missing columns from df1: df_replace = pd.concat([df1[k].replace(val, regex=True) for k, val in d.items()], axis=1).join(df1.state)
13. Turn Nested JSON to Pandas Dataframe (57200878)	SE	JSON.parse(jsonString) is a pure JavaScript approach so long as you can guarantee a reasonably modern browser.
	Google	Use: df2=df['location'].apply(lambda x: pd.DataFrame(x)) pd.concat([df[df.columns[:-1]],df2],axis=1) Also you can try json_normalize from pandas.io.json import json_normalize df=json_normalize(json_documents)

	Ours	Use json_normalize:
		import pandas as pd
		from pandas.io.json import json_normalize
		from pathlib import Path
		import json
		# path to file
		p = Path(r'c:\path_to_file\test.json')
		# read json
		with p.open('r', encoding='utf-8') as f:
		  data = json.loads(f.read())
		# create dataframe
		df = json_normalize(data, record_path='location', meta=['id', 'name'])

Unanswered Questions User Study		
14. how can i run python script on multiple files in a folder and write the output (57209151)	SE	shutil has many methods you can use. One of which is: from shutil import copyfile copyfile(src, dst) Copy the contents of the file named src to a file named dst. The destination location must be writable; otherwise, an IOError exception will be raised. If dst already exists, it will be replaced. Special files such as character or block devices and pipes cannot be copied with this function. With copy, src and dst are path names given as strings. If you use os.path operations, use copy rather than copyfile. copyfile will only accept strings.

	Google	indent the code that does the dataframe transformation and include it in the for loop like this:
		import glob
		import pandas as pd
		import numpy as np
		files = glob.glob('folder/*.csv')
		for file in files:
		df = pd.read_csv(file)
		#make conversion
		df['Time taken'] = pd.to_datetime(df['Time taken'])
		df['Time taken'] = df['Time taken'].dt.hour + df['Time taken'].dt.minute / 60
		#output file
		df.to_csv('updated_{}'.format(file), index = False)

	Ours	for f in *.csv; do python playlist.py "$f" "${f%.csv}list.txt" done Will that do the trick? This will put foo.csv in foolist.txt and abc.csv in abclist.txt. Or do you want them all in the same file?
15. How can I make my script wait for input for a set amount of time ()57245591	SE	Enums have been added to Python 3.4 as described in PEP 435. It has also been backported to 3.3, 3.2, 3.1, 2.7, 2.6, 2.5, and 2.4 on pypi. For more advanced Enum techniques try the aenum library (2.7, 3.3+, same author as enum34. Code is not perfectly compatible between py2 and py3, e.g. you'll need __order__ in python 2). To use enum34, do $ pip install enum34 To use aenum, do $ pip install aenum Installing enum (no numbers) will install a completely different and incompatible version. ….

	Google	Looks like you want something like this: this is the most flexible way to expect something to happen multiple times, and we don't have to care about exactly how many times it happens.
		expect -c '
		set timeout 1200 ;# 20 minutes
		spawn sh ./setupapp.sh
		expect {
		"PRESS <ENTER> TO CONTINUE:" {
		send "\r"
		exp_continue}
		"Waiting here:" {
		send "\r"
		exp_continue}
		timeout {
		error "nothing happened after $timeout seconds" }
		"PRESS <ENTER> TO Exit Installation:" {
		send "\r"}}
		expect eof
		'
		That expect command waits for one of four things to happen. For the first 2, hit enter then keep waiting for another event. I added the "timeout" event in case you want to do something special there.
		The "press enter to exit" block does not call "exp_continue". After it sends the carriage return, the enclosing expect command ends and we then wait for eof.

	Ours	Do multi-threading. Threads allow you do some tasks simultaneously. I suggest you do the input-fetching operation as a thread and a timer operation on another thread. So now, you can check the timer while fetching the input and pause/resume the fetch when the timer reaches a particular limit. In case you need references on multi-threading: Multi-Threading tutorial in tutorialspoint

16 search excel file for matching text word, get row contents (57256050)	SE	>>> import os
		>>> os.path.abspath("mydir/myfile.txt")
		'C:/example/cwd/mydir/myfile.txt'
		Also works if it is already an absolute path:
		>>> import os
		>>> os.path.abspath("C:/example/cwd/mydir/myfile.txt")
		'C:/example/cwd/mydir/myfile.txt'

	Google	For an approach to searching I suggest you look at the answer to this question. Create a dictionary of terms to search in a single pass and use it as often as you like. However, you might also like to know that openpyxl also supports conditional formatting so that you can delegate the formatting to Excel. See the very bottom of the examples in the documentation. Used this myself for the first time last week. The Excel API is stupid as usual but you could abstract it and add all kinds of formatting to the file without ever searching yourself.

	Ours	This should work. You can't assign 5 strings to a single variable, without using a list or some other data type. You can however check to see if the third cell's value (i[2] - here) is equal to any of the strings you're looking for ("string1" - "string5" - here).
		import xlrd
		sheet_data = []
		wb = xlrd.open_workbook(Path_to_xlsx)
		p = wb.sheet_names()
		for y in p:
		sh = wb.sheet_by_name(y)
		for rownum in xrange(sh.nrows):
		sheet_data.append((sh.row_values(rownum)))
		found_list = []
		rows_to_be_saved = []
		for i in sheet_data:
		if i[2] == "string1" or i[2] == "string2" or i[2] == "string3" or i[2] == "string4" or i[2] == "string5":
		found_list.append(i)
		else:
		rows_to_be_saved.append(i)
		text_file = open("Output.txt", "w")
		text_file.write(found_list)
		text_file.close()
		Your output written to the text file "Output.txt" will be comma separated as the rows in your excel are read into python as tuples in a list.

Unanswered Questions User Study		
17. How to change font face and size of content of a HTML document in python? (57325394)	SE	shutil has many methods you can use. One of which is: from shutil import copyfile copyfile(src, dst) Copy the contents of the file named src to a file named dst. The destination location must be writable; otherwise, an IOError exception will be raised. If dst already exists, it will be replaced. Special files such as character or block devices and pipes cannot be copied with this function. With copy, src and dst are path names given as strings. If you use os.path operations, use copy rather than copyfile. copyfile will only accept strings.
	Google	To rectify this send tag inside the empty string. It will work!!!. Instead If we send inside empty string, inside the .doc file on the libre writer it will show Field Shadings while the document if opened in edit mode. That's the main issue, that's why I asked this question here.
	Ours	from docx import Document from docx.shared import RGBColor document = Document() run = document.add_paragraph() binary_run = run.add_run('your_binary_code') binary_run.font.color.rgb = RGBColor(rgb_color_code_goes_here) cmplt_run = binary_run.add_run('rest of the text goes here') This will change the font color of ur binary code to the color code you provide. Refer python-docx documemtation to understand more.
18. Matrix multiplication in Python without Numpy (57325193)	SE	If the reason you're checking is so you can do something like if file_exists: open_it(), it's safer to use a try around the attempt to open it. Checking and then opening risks the file being deleted or moved or something between when you check and when you try to open it. If you're not planning to open the file immediately, you can use os.path.isfile , Return True if path is an existing regular file. This follows symbolic links, so both islink() and isfile() can be true for the same path. import os.path os.path.isfile(fname) …
	Google	The Numpythonic approach: (using numpy.dot in order to get the dot product of two matrices) In [1]: import numpy as np In [3]: np.dot([1,0,0,1,0,0], [[0,1],[1,1],[1,0],[1,0],[1,1],[0,1]]) Out[3]: array([1, 1]) The Pythonic approach: The length of your second for loop is len(v) and you attempt to indexing v based on that so you got index Error . As a more pythonic way you can use zip function to get the columns of a list then use starmap and mul within a list comprehension: In [13]: first,second=[1,0,0,1,0,0], [[0,1],[1,1],[1,0],[1,0],[1,1],[0,1]] In [14]: from itertools import starmap In [15]: from operator import mul In [16]: [sum(starmap(mul, zip(first, col))) for col in zip(*second)] Out[16]: [1, 1]

	Ours	Without numpy, you can write yourself a function for the dot product which uses zip and sum.
		
		def dot(v1, v2):
		  return sum(x*y for x,y in zip(v1,v2))
		
		>>> dot([1,2,3], [4,5,6])
		32

19. Unable to run pandas python script via batch file (57296324)	SE	Another useful tip is to use %* to mean "all". For example: echo off set arg1=%1 set arg2=%2 shift shift fake-command /u %arg1% /p %arg2% %* When you run: test-command admin password foo bar the above batch file will run: fake-command /u admin /p password admin password foo bar I may have the syntax slightly wrong, but this is the general idea.
	Google	What you are missing is probably the conda environment activation in your batch file. call activate [your_env_name] python your_script_name.py call conda deactivate
	Ours	Regarding your batch file, I would suggest perform these changes: You were missing a space inbetween the path and the %*, also enclose the path in quotes to prevent it from breaking when spaces are in the path @echo off py.exe "C:\MyPythonScripts\program.py" %1 %2 %3 %4 %5 %6 %7 %8 %9 pause
20. Python: How to import package correctly in python? (57147927)	SE	For dictionaries x and y, z becomes a shallowly merged dictionary with values from y replacing those from x. In Python 3.5 or greater: z = {**x, **y} … Resources on Dictionaries.
	Google	"I have a medium size Python application with modules files in various subdirectories.” Good. Make absolutely sure that each directory include a __init__.py file, so that it's a package. "I have created modules that append these subdirectories to sys.path” Bad. Use PYTHONPATH or install the whole structure Lib/site-packages. Don't update sys.path dynamically. It's a bad thing. Hard to manage and maintain. Bad. Use PYTHONPATH or install the whole structure Lib/site-packages. Don't update sys.path dynamically. It's a bad thing. Hard to manage and maintain. …… My current project has 100's of modules, a dozen or so packages. Each module imports just what it needs. No magic.

Unanswered Questions User Study		
	Ours	Since there are already many answers on SO for this*, I will focus on question (2). About what is a better code organization: … The (relative) import is done as follows, from inside module_2: from ..pkg1 import module1 as m1 Alternatively, you can use absolute imports, which refer to the top package name: from top_pkg_name.pkg1 import module1 as m1 In such an organization, when you want to run any module as a script, you have to use the -m flag: python -m top_pkg_name.pkg1.module1
21How to replace a value in a dataset django (57293384)	SE	For dictionaries x and y, z becomes a shallowly merged dictionary with values from y replacing those from x. In Python 3.5 or greater: z = {**x, **y} … Resources on Dictionaries.
	Google	You can update all the records in the queryset with qs.update(active=False) Please refer to the official Django documentation for more info
	Ours	You can 'mock' that property using the mock library from mock import patch, PropertyMock @patch.object(Test, 'haveCode', new_callable=PropertyMock) def myTest(test_haveCode_mock): test_haveCode_mock.return_value = True checkTest = Test() assertEqual(checkTest.haveCode, True) patch.stopall() # when you want to release all mocks
22. handling masked numpy array with ray (57293187)	SE	Yes, here is the answer given a NumPy array, array, and a value, item, to search for: itemindex = numpy.where(array==item) The result is a tuple with first all the row indices, then all the column indices. For example, if an array is two dimensions and it contained your item at two locations then array[itemindex[0][0]][itemindex[1][0]] would be equal to your item and so would array[itemindex[0][1]][itemindex[1][1]]
	Google	I think the problem is that you set all those pixels to zero, then try to get statistics from them. Instead, realize that test1[image < 100] only refers to those pixels below the threshold... so I think you can just get your statistics from that, e.g. with np.std(test1[image < 100]). You might like to take a look at scikit-image, which contains lots of tools for thresholding, dealing with binary images, using those as masks (which is essentially what you're doing), etc.
	Ours	You can use np.broadcast_arrays: a475 = np.stack(np.broadcast_arrays(a75, a71, a15, a11)) Note that this creates views of the original arrays, so it should not entail any extra memory usage.
23. How to read a bit field integer in python? (57152181)	SE	For dictionaries x and y, z becomes a shallowly merged dictionary with values from y replacing those from x. In Python 3.5 or greater: z = {**x, **y} … Resources on Dictionaries.

	Google	The following code will load the requested portions of the binary number into the fields:
		class Register(object):
		def __init__(self,x):
		self.fieldwidths = [6,12,6,4,12,8,16]
		……
		Which will result in:
		BitField1 = 0b111011
		BitField2 = 0b111011101110
		……
		The results may not be what you wanted because of the fact that the data you provided is not 64 bits but rather 128 bits, which would mean that the 64 most significant bits of the input data will be ignored by the program.

	Ours	In Python 3 you can use something like this: int.from_bytes(byte_string, byteorder='little')
24. How do I enable Windows' scp for use from Python? (57290518)	SE	New answer, powershell Get-Process -Id (Get-NetTCPConnection -LocalPort portNumber).OwningProcess Old answer, cmd C:\> netstat -a -b (Add -n to stop it trying to resolve hostnames, which will make it a lot faster.) Note Dane's recommendation for TCPView. It looks very useful!
	Google	Try the Python scp module for Paramiko. It's very easy to use. See the following example: import paramiko from scp import SCPClient def createSSHClient(server, port, user, password): client = paramiko.SSHClient() client.load_system_host_keys() client.set_missing_host_key_policy(paramiko.AutoAddPolicy()) client.connect(server, port, user, password) return client ssh = createSSHClient(server, port, user, password) scp = SCPClient(ssh.get_transport()) Then call scp.get() or scp.put() to do SCP operations.

Unanswered Questions User Study		
	Ours	Try the Python scp module for Paramiko. It's very easy to use. See the following example: import paramiko from scp import SCPClient def createSSHClient(server, port, user, password): client = paramiko.SSHClient() client.load_system_host_keys() client.set_missing_host_key_policy(paramiko.AutoAddPolicy()) client.connect(server, port, user, password) return client ssh = createSSHClient(server, port, user, password) scp = SCPClient(ssh.get_transport()) Then call scp.get() or scp.put() to do SCP operations.
25. How to scrape _doPostback ajax website with next_page link python (57289799)	SE	shutil has many methods you can use. One of which is: from shutil import copyfile copyfile(src, dst) Copy the contents of the file named src to a file named dst. The destination location must be writable; otherwise, an IOError exception will be raised. If dst already exists, it will be replaced. Special files such as character or block devices and pipes cannot be copied with this function. With copy, src and dst are path names given as strings. If you use os.path operations, use copy rather than copyfile. copyfile will only accept strings.
	Google	Webkit based browsers (like Google Chrome or Safari) has built-in developer tools. In Chrome you can open it Menu->Tools->Developer Tools. The Network tab allows you to see all information about every request and response: In the bottom of the picture you can see that I've filtered request down to XHR - these are requests made by javascript code. Tip: log is cleared every time you load a page, at the bottom of the picture, the black dot button will preserve log. After analyzing requests and responses you can simulate these requests from your web-crawler and extract valuable data. In many cases it will be easier to get your data than parsing HTML, because that data does not contain presentation logic and is formatted to be accessed by javascript code. Firefox has similar extension, it is called firebug. Some will argue that firebug is even more powerful but I like the simplicity of webkit.
	Ours	Check out scrapy_splash, it's a rendering service for scrapy, that will allow you to crawl javascript based web sites. You can also create your own downloader middleware and use Selenium with PhantomJS (example). The downside of this technic is that you lose the concurrency provided by scrapy. Anyway, I think splash is the best way to do this. Hope this helps.